CHOREOGRAPHY DESCRIPTION IN SERVICE-ORIENTED ARCHITECTURE

Martin Uřídil

Bachelor Degree Programme (3), FIT BUT

E-mail: xuridi00@stud.fit.vutbr.cz

Supervised by: Marek Rychlý

E-mail: rychly@fit.vutbr.cz

Abstract: This paper describes how services can be composed in service-oriented architecture. Particularly it is focused on choreography of services and possibilities of its description. The abstract explains the advantages of formal description using the process algebra and demonstrates how it can be leveraged for validating and testing choreographies. Last part of this paper describes whole development cycle from conceptual model to implementation and deployment of sample service oriented architecture.

Keywords: SOA, choreography, π-calculus, process algebra, web-services

1. ÚVOD

Servisně orientovaná architektura (dále SOA) nabývá v dnešním světě informačních technologií stále většího významu. Implementace tohoto přístupu k pojetí podnikové informační struktury obvykle vychází ze samotného popisu byznys procesů v tomto podniku. Proto umožňuje lépe takové procesy podporovat a automatizovat. SOA umožňuje nejen automatizaci procesů uvnitř podniku, ale především podporu procesů, kde jsou účastníky různé společnosti a procesy tak probíhají v prostředí, které nevlastní ani jeden z partnerů, viz [1].

Cílem této práce je představit, jak mohou být webové služby v systému komponovány a jak je řízena jejich vzájemná komunikace. Očekávaným přínosem tohoto článku a navazující bakalářské práce je analýza popisů choreografie služeb a uvedení předností formálních popisů. Dále pak demonstrace využití formálního popisu modelů v procesní algebře π-kalkulus při vytváření šablon kódu a validaci choreografie. Je očekáván také didaktický přínos, kdy na ukázkové aplikaci práce demonstrujeme metodiku zavádění formálního popisu choreografie do modelování SOA.

2. KOMPOZICE SLUŽEB V SOA

Servisní architektura zahrnuje vždy jednotlivé služby a jejich vzájemnou komunikaci. Službou rozumíme aplikaci nebo její část vykonávající určitou, předem danou činnost. Komunikací rozumíme výměnu zpráv mezi službami, což je podrobněji rozvedeno v [2], kapitola 3.

Podle toho, zda je řízení komunikace delegováno na nadřazenou službu nebo je rovnoměrně rozděleno mezi účastníky komunikace, rozdělujeme kompozice na orchestrace a choreografie, více v [1].

2.1. ORCHESTRACE

Orchestrace se využívá především pro řízení komunikace mezi službami v rámci jednoho subjektu (podniku). Je to z toho důvodu, že všechny služby jsou ve vlastnictví jednoho subjektu a ten má proto prostředky, aby komunikaci řídil centralizovaně, například tomu určenou službou.

2.2. CHOREOGRAFIE

Zatímco orchestrace popisuje sekvenci kroků v rámci procesu, choreografie se soustředí spíše na cíl spolupráce a výměnu zpráv mezi jednotlivými službami. Své využití nachází především při komunikaci přesahující hranice organizace. Příkladem mohou být objednávkové systémy velkých společností, které musí být schopné samostatně řešit objednávky od dodavatelů. V takových situacích není možné komunikaci řídit centralizovaně, protože neexistuje společný vlastník všech služeb.

Řešením je choreografie, kde každá služba v systému je chápána jako autonomní a znalá pouze své části úlohy. Proto můžeme v choreografiích vidět silnou podobnost s multiagentními systémy.

3. MOŽNOSTI POPISU CHOREOGRAFIÍ

Popisy choreografií jednoznačně definují posloupnost výměny zpráv mezi množinou služeb. Jejich cílem je:

- Podpořit porozumění mezi účastníky komunikace
- Zajistit spolupráci webových služeb
- Umožnit verifikaci implementace oproti návrhu modelu
- Přispět k robustnosti systému
- Umožnit automatické generování kostry zdrojového kódu

Popisů choreografií existuje relativně velké množství, některé k dosažení stanovených cílů přispívají více, některé méně.

3.1. STANDARDY PRO POPIS CHOREOGRAFIÍ

Vzhledem k tomu, že architektura SOA umožňuje spolupráci systémů nezávislých na platformě i vlastníkovi, jsou zapotřebí určité standardy, které budou účastníci dodržovat.

Organizace OASIS vydala za tímto účelem BPEL4CHOR [3], což je rozšíření jazyka pro popis byznys procesů ve webových službách BPEL o popis choreografie. Na rozdíl od čistého BPEL, jazyk BPEL4CHOR není vykonatelný.

Druhou, neméně významnou, organizací v této oblasti je W3C, která pro popis choreografií navrhla WS-CDL, jazyk založený na XML a inspirovaný procesní algebrou π-kalkulus (bude popsáno dále).

Ani jeden z těchto jazyků, však neposkytuje příliš dobrou podporu pro validaci choreografií. Proto jsou jazyky používány spíše jako modely, které jsou předlohou k implementaci.

3.2. FORMÁLNÍ MODELY PRO POPIS CHOREOGRAFIÍ

Obecně platí, že lepší podporu pro validaci a verifikaci choreografií poskytují popisy založené na formálním modelu. Jako formální modely se nejvíce hodí popisy pro konkurentní procesy, např. Petriho sítě nebo procesní algebry [4].

Procesní algebry obsahují dva druhy prvků. Prvním jsou procesy, které mohou být v SOA představovány samotnými službami. Obecně to jsou komunikující entity. Druhým jsou jména neboli kanály zastupující prvky, pomocí nichž probíhá komunikace. V některých implementacích mohou být kanály i předmětem komunikace (například v SOA jsou zprávy kanálem).

V této práci byla vybrána procesní algebra π-kalkulus jako formální model pro ukázkovou choreografii. Více o π-kalkulu v [5].

4. IMPLEMENTACE UKÁZKOVÉ ARCHITEKTURY SOA

Implementace ukázkové architektury zahrnuje celý vývojový proces, od návrhu konceptuálního modelu systému, přes testovatelný abstraktní model vyjádřený pomocí π-kalkulu až po finální implementaci a nasazení.

4.1. NÁVRH CHOREOGRAFIE S VYUŽITÍM Π-KALKULU

Při návrhu samotné choreografie je využíván nástroj s názvem PI4SOA [6], který je postaven nad formalismem π-kalkulu a poskytuje grafické vývojové prostředí pro návrh, validaci a export choreografií do jazyka WS-CDL a BPEL. Tento nástroj zároveň umožňuje na základě tzv. scénářů choreografie testovat a validovat. Nástroj je vytvořen v jazyce Java a prostředí Eclipse.

Vzhledem k mému záměru implementovat řešení v prostředí .NET, kde PI4SOA nemá žádnou podporu, je tento nástroj použit pouze pro získání popisu ve WS-CDL. Nicméně při testování a validaci choreografie je přínos formálního modelu obrovský.

4.2. IMPLEMENTACE SLUŽEB

Na základě vygenerovaného popisu v jazyce WS-CDL byla vytvořena nejprve kostra architektury a následně implementována vnitřní funkcionalita webových služeb. Pro komunikaci služeb pomocí zpráv SOAP se již naplno využívá možností frameworku .NET, konkrétně jeho knihovny pro komunikaci, WCF.

5. ZÁVĚR

Tato práce představila možnosti popisu kompozice služeb v servisně orientovaných architekturách, především se pak soustředila na popisy choreografií. V této oblasti ukázala přednosti formálních modelů při validaci a testování architektury. Druhá část práce pak demonstruje využití formálního popisu v návrhovém procesu ukázkové architektury. Pro popis choreografie služeb se používá procesní algebra π-kalkulus. V nástroji PI4SOA se poté provádí základní testování a validace navržené choreografie. Velkým přínosem tohoto přístupu je dosažení bezpečného řešení ještě před jeho implementací.

Pro budoucí vývoj by mohla být zajímavá implementace validační a testovací komponenty pro choreografie v prostředí .NET. Nástroj by měl kromě validací podporovat i generování rozhraní služeb a šablon jejich kódu. Taková komponenta v nabídce produktů .NET momentálně chybí a její implementace by určitě našla mnoho uplatnění.

Tento příspěvek byl podpořen výzkumným záměrem č. MSM 0021630528.

REFERENCE

[1] Erl, T.: SOA Principles of Service Design, Upper Saddle River, Prentice Hall 2008, ISBN 978-013-2344-821

[2] Erl, T.: SOA Kompletní průvodce, Computer Press, Brno, 2009, ISBN 978-80-251-1886-3

[3] BPEL4CHOR [online]. Dostupné z URL: http://www.bpel4chor.org/ (únor 2011)

[4] Aalst W.M.P.: Pi calculus versus Petri nets [online]. Dostupné z URL: http://is.tm.tue.nl/research/patterns/download/pi-hype.pdf (únor 2011)

[5] Rychlý, M.: Formální specifikace architektur informačních systémů [online]. Dostupné z URL: http://www.fit.vutbr.cz/~rychly/public/docs/formal-architectures/xhtml/formarch.xhtml#sec-procalg (březen 2011)

[6] Pi4soa [online]. Dostupné z URL: http://sourceforge.net/apps/trac/pi4soa/wiki (únor 2011)